TOOL FOR DISPLAYING JMX MONITORING INFORMATION

TECHNICAL FIELD

[1] The present invention relates generally to monitoring the status of components in a J2EE environment and, in particular, to improving the efficiency with which status information is obtained and displayed.

BACKGROUND ART

[2] Java Management Extensions (JMX) application programming interface (API) provides a means for asynchronously determining the status of system components (such as servers, applications, processes or other resources) in a Java Enterprise Edition (J2EE) server. The status monitoring API requires a user to perform complex queries in order to obtain the status and consequently, it is advantageous to provide a simple client which allows a user to visualize the status of large groups of components. Such a client may consist of a web page containing a series of images corresponding to the states of the various components.

[3] JMX queries are executed asynchronously and may not be completed within any specific period of time. However, the content of a web page typically loads and displays serially. Thus, a JMX query may not begin until the previous query has been completed. Because some queries may take a long time to complete while other may take only a short time, a single, long-running query may block the completion and display of other, shorter queries.

[4] Fig. 1 is a block diagram of a prior art process 100 of monitoring the status of components. A user at a client device, such as a web-based administrative console 102, initiates JMX queries to determine the status of several server components 110 and 120. For clarity, only two components are illustrated; many more may be included in the system and the queries. The queries are first transmitted to a servlet 104 which processes the first query 106 to determine the status of the first server 110. The first server 110 responds after which the servlet 104 processes the second query 108 to determine the status of the second server 120. Only when the

Docket: RSW920030279US1
Express Mail Label: EV303489949US

responses from all of the servers has been received by the servlet 104 does the servlet 104 provide them to the browser running in the client 102.

[5] Moreover, the user may be interested in only a subset of the objects for which the status is presented or may be interested in information other than status, such as a simple list of application servers. If there are ten such servers and each JMX query takes up to three minutes to complete, the user may have to wait for as long as thirty minutes to see the results, even though the specific information desired by the user may be available almost immediately.

[6] Consequently, a need remains for improving the efficiency with which monitoring information is provided to a user.

SUMMARY OF THE INVENTION

[7] The present invention provides method, system and computer program product to allow the status of components obtained through JMX queries to be loaded asynchronously in web content such that the status value of each component is displayed as soon as it becomes available, without having to wait for the completion of any other query.

BRIEF DESCRIPTION OF THE DRAWINGS

[8] Fig. 1 is a block diagram of a prior art component monitoring system;

[9] Fig. 2 is a block diagram of a component monitoring system of the present invention;

[10] Fig. 3 is a flow chart of the component monitoring system of the present invention; and

[11] Fig. 4 is an exemplary screen shot of the display of component status information obtained with the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[12] Fig. 2 is a block diagram of a component monitoring system 200 of the present invention. The system 200 includes a web-based administrative console 202 running a web browser 203. And As before, only two components 210 and 220 are

shown although more may be included. Referring also to the flow chart of Fig. 3, a user at the console 202 requests a web page containing the status of any number of objects or components, such as the servers 210 and 220 (step 300). The status may include whether the component is stopped, started, in error, etc. The user may also request other information pertaining to the components 210 and 220. A servlet 204 is called (step 302) to dynamically generate an HTML file (step 304) which ultimately will be used to display the status information on the console 202. Although in the prior system illustrated in Fig. 1 a servlet 104 would determine each component's status and include in the HTML file an image corresponding to the status, in the present invention 200 the servlet 204 instead includes an image tag (step 306) which points to another servlet. Moreover, the image tag includes enough information to perform a JMX query of the corresponding component. For example, an exemplary image tag might comprise:

<image src="/servlet/StatusServlet?component=Server&name=MyServer"/>

which calls a servlet named StatusServlet and returns an image representing the status of a component of type "Server" with the name "MyServer".

[13] The web page defined by the HTML file is loaded by the browser 203 running in the console (step 308) and places status requests for each of the image tags (step 310). These requests are placed in parallel. The requests are transmitted to each server 210 and 220 (step 312) but processed by copies or instances of the second servlet 212 and 222 (step 314) (StatusServlet in the above example). Each instance of the servlet 212, 214 uses the information provided in the request and generates a JMX query on the corresponding server (step 316). After receiving a response to the query in the form of a value representing the status of the component (step 318), each servlet 212, 214 processes the response and transmits it to the browser 203 (step 320). The processed response includes an image corresponding to the value returned from the JMZ query. The processed response also preferably includes a directive to prevent the browser from caching the response.

[14] The browser 203 receives the response (step 322) which appears as a standard displayable image. As each response is received by the browser 203, it is displayed (step 324) for the user even if other queries are still being processed. Referring

again to the original example, if there are ten components and each JMX query takes up to three minutes to complete, the user will only have to wait approximately three minutes to see results, a significant improvement over the prior sequential method. Fig. 4 illustrates an exemplary screen shot of a browser window displaying status information received for several resources.

[15] The objects of the invention have been fully realized through the embodiments disclosed herein. Those skilled in the art will appreciate that the various aspects of the invention may be achieved through different embodiments without departing from the essential function of the invention. The particular embodiments are illustrative and not meant to limit the scope of the invention as set forth in the following claims.